まい CAN 取扱説明書

株式会社 北斗電子

(REV.1.0.4.0)

目　　次

1. はじめに 2
2. 製品内容 3
3. 概要 4
4. 製品仕様 5
5. 「まい CAN」を組み込む方法（開発編） 5
6. 「まい CAN」を組み込む方法（運用編） 5
7. API 関数（ユーザーインターフェイス） 6
   7.1. API 関数一覧 6
   7.2. ライブラリ状態遷移 7
   7.3. API 関数詳細 8
8. 付録 17
9. 「まい CAN」と「まい CANLite」の違いについて 19
10. 著作権・免責について 19

1. はじめに

今日では組込みシステムは様々な機器に採用され、組込みシステムプログラムの需要が益々増しています。

組込みシステムを構成する上で必要な要素の一つとして様々な通信方式があり、例を上げるとRS-232C、Ethernet、LIN、CAN、IIC などがあります。

CAN は ISO11898 および ISO11519 等で標準化されたシリアル通信プロトコルとなっておりますが、CAN 規格や CAN 規格に対応した CPU のモジュール設定などについて十分に理解するのに多くの時間を要します。

「まい CAN」はそういった難しい知識をアプリケーションに意識させることなく、CAN 通信を実現するお手軽なライブラリソフトウェアです。

この取扱説明書はソフトウェアの使用方法について説明しています。

正しくご使用していただくために、この取扱説明書をよくお読みください。

2. 製品内容

次のものが全てそろっているかどうかをご確認ください。

- ＣＤ.....1 枚
- ソフトウェア使用許諾書.....1 枚

「MYCAN_XXXX.lib」の XXXX は製品 ID を意味しております。

CD 内容

MYCAN

manual　取扱説明書(PDF 形式)

sample　サンプルソフト一式

software　ライブラリー式
・インクルードヘッダ
・ライブラリ

ソフトウェア使用許諾書(PDF 形式)

3. 概要

「まいCAN」は、CAN通信を意識しないでCAN通信を行うためのソフトウェアライブラリです。
「まいCAN」のAPI関数をコールするだけで、CAN通信を簡単に行うことができます。
API関数の詳細については、7章のAPI関数（ユーザーインターフェイス）をご参照ください。

「まいCAN」を使用したCAN通信の動作概念図を図 1に示します。

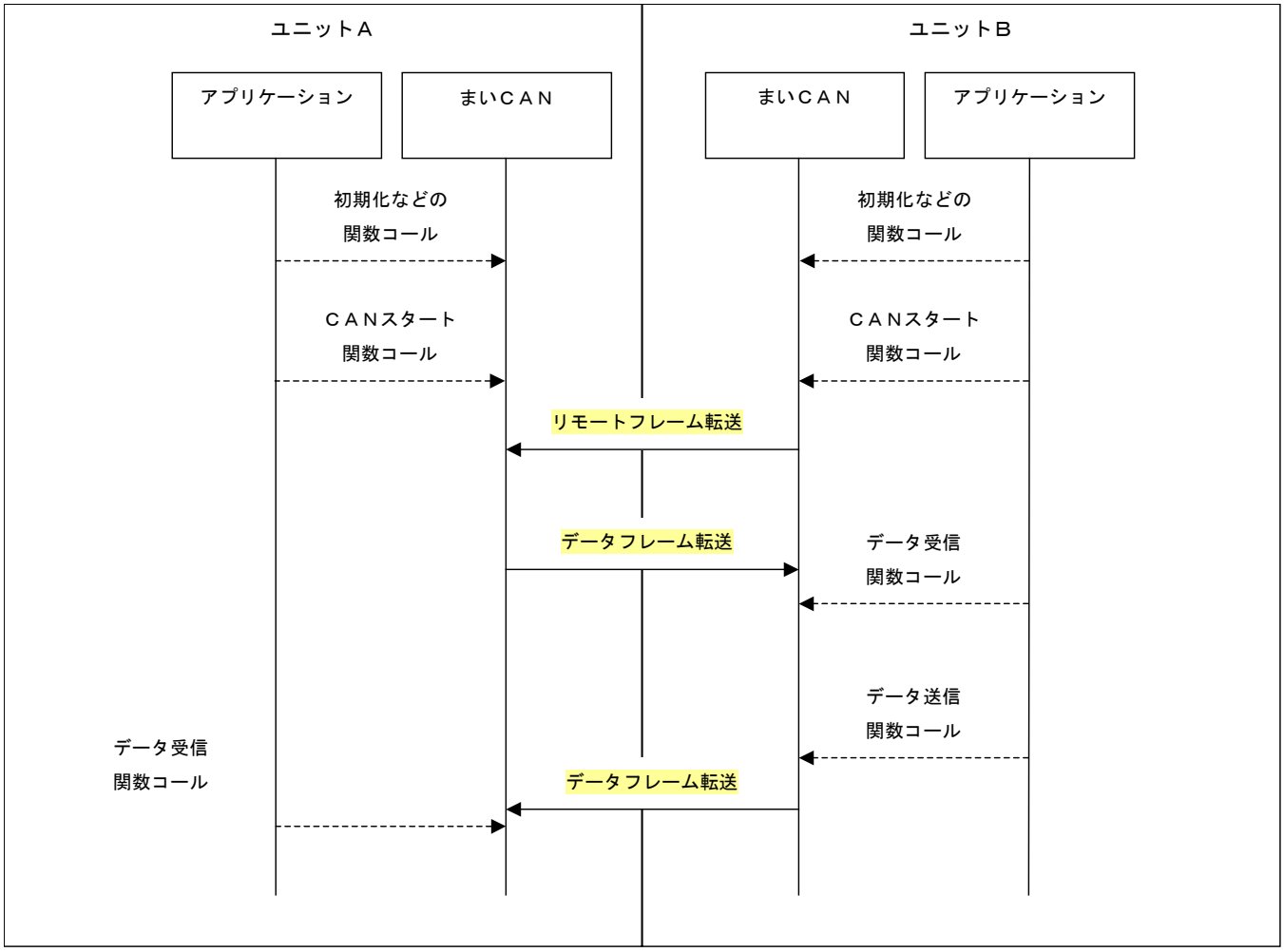


図 1「まいCAN」動作概念図

4. 製品仕様

本製品は、CAN 規格 2.0B に対応しているソフトウェアライブラリとなります。
「まい CAN」では、以下のサイズを必要とします。

セクション名	サイズ
プログラム領域	15KByte
定数領域	300Byte
未初期化データ領域	1.5KByte
スタック	64Byte

5. 「まい CAN」を組み込む方法（開発編）

- API 関数をご使用になられるアプリケーションのファイルに、「can_api.h」をインクルードするように実装してください。
- アプリケーションの仕様に合わせて7.3章に記載しています API 関数をご使用ください。
- コンパイルを行う際には、CD に入っています「can_api.h」を開発環境（インクルードパス上）にコピーしてください。
- リンクを行う際には、CD に入っています「MYCAN_XXXX.lib」を開発環境（入力ライブラリファイル指定パス上）にコピーしてください。
  お使いのコンパイル/リンク環境によって以下の形式のものをご利用下さい。
  ・バンク切換なしの ELF 形式: MYCAN_XXXX_elf.lib
  ・バンク切換ありの ELF 形式: MYCAN_XXXX_resbank.lib(バンク切換に対応したマイコンのみ)

6. 「まい CAN」を組み込む方法（運用編）

(1) 5章にて開発しました CAN ソフトウェアをお手持ちのライターでターゲット基板に書き込んでください。
(2) ターゲット基板と CAN 通信可能な機器とをケーブルで接続し、ターゲット基板の電源を投入してください。
(3) 完了

7. API 関数（ユーザーインターフェイス）

「まい CAN」はライブラリ形式となっています。
アプリケーションからは、ライブラリをリンクすることで、ライブラリの API 関数をコールすることができます。

【注意事項】
- 割り込み処理を行っておりますので、割り込みベクタテーブルにCAN割り込み関数「MyCanInterruptRm」「MyCanInterruptSIe」「MyCanInterruptTimer」をご登録ください。
  また、割り込みマスク解除処理を行ってください。
  H8S/2556の場合、ライブラリ内で登録しておりますので登録の必要はございません。
- CANバス送信用端子（CTxx）及びCANバス受信用端子（CRxx）がマルチプレクス端子の場合、ハードウェア仕様に合わせたポート設定を行ってください。

7.1. API 関数一覧

API 関数一覧を表 7-1、表 7-2に示します。CAN モジュールが複数搭載されているターゲットは、API 関数末尾に「_CHn」（n：チャネル番号）のついたものが関数名となります。

表 7-1 API 関数一覧

関数名	説明
MyCanInit	CAN ライブラリ初期化
MyCanSetBitRate	ビットレート設定
MyCanSetMailBoxInfo	メールボックス情報設定
MyCanEnable	CAN 通信実行
MyCanDisable	CAN 通信停止
MyCanSetSendData	送信データ設定
MyCanGetSendDataStatus	データ送信状態取得
MyCanGetRecvData	データ受信
MyCanGetStatus	ユニット状態取得
MyCanGetLibStatus	CAN ライブラリ状態取得
MyCanSetPriority	割り込み優先度設定

表 7-2 割り込み処理関数一覧

関数名	説明
MyCanInterruptRm	データフレーム／リモートフレーム受信割り込み
MyCanInterruptSIe	データフレーム／リモートフレーム送信完了割り込み
MyCanInterruptTimer	タイマー割り込み

7.2. ライブラリ状態遷移

「まい CAN」はアプリケーションからの関数コールにより状態が遷移します

図 7-1に状態遷移図を示します。また、各状態時実行可能な関数一覧を表 7-3に示します。

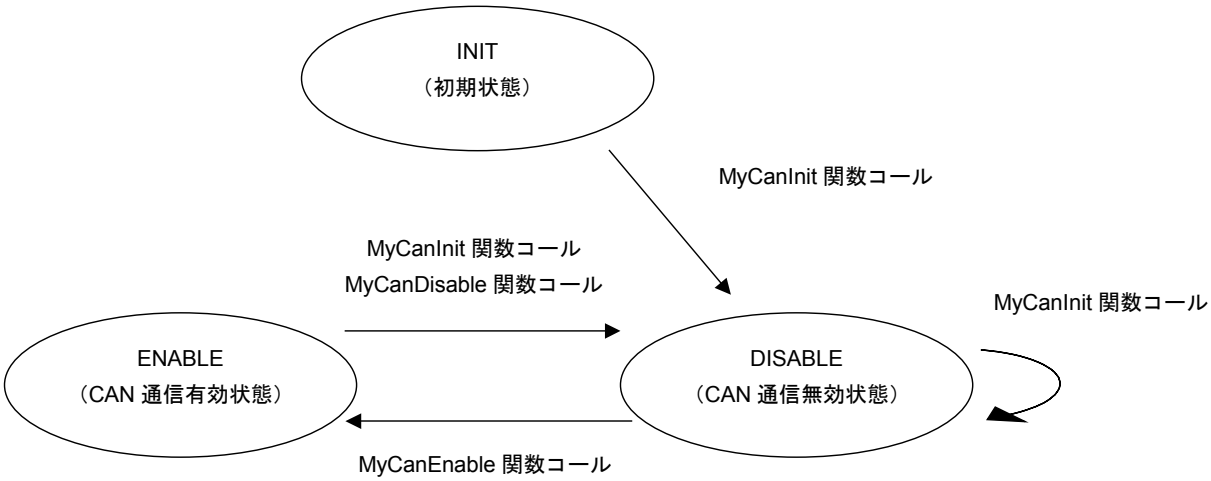


図 7-1　状態遷移図

表 7-3　各状態での関数実行一覧

関数名	CAN ライブラリ状態		
	INIT	DISABLE	ENABLE
MyCanInit	○	○	○
MyCanSetBitRate	×	○	×
MyCanSetMailBoxInfo	×	○	×
MyCanEnable	×	○	×
MyCanDisable	×	×	○
MyCanSetSendData	×	×	○
MyCanGetSendDataStatus	×	○	○
MyCanGetRecvData	×	×	○
MyCanGetStatus	×	○	○
MyCanGetLibStatus	○	○	○
MyCanSetPriority	×	○	×

○　：実行可能　　×：実行不可能

7.3. API 関数詳細

MyCanInit

CAN 初期化

書式	void MyCanInit (void)
引数	なし
戻り値	なし
解説	「まい CAN」を初期化します。 本関数実行時には、ビットレート設定、メールボックス設定などの設定情報が全て初期化されます。 本関数未実行時には CAN 通信が動作致しませんので、起動時に実行してください。

MyCanSetBitRate

ビットレート設定

書式	signed short MyCanSetBitRate (unsigned long rate float fclk)	
引数	unsigned long rate	｜　ビットレート（Kbps）（1～1000）
	float fclk	｜　周辺クロック（MHz）小数点第 2 位まで有効
戻り値	OK(0)	：正常終了
	NG(-1)	：異常終了 ・ ビットレートが範囲外 ・ 周辺クロックが範囲外 ・ CAN ライブラリ状態が DISABLE 以外 ・ 誤差率 1%を超過している場合
解説	CAN の通信レートを設定します。 対応マイコンにより周辺クロックに設定可能な範囲が異なります。 ・ SH7216：20.00MHz～50.00MHz ・ SH7262/7264：6.7MHz～36.00MHz ・ H8S/2556:8.0MHz～20.00MHz ・ SH72A2：8.0MHz～50.00MHz	

MyCanSetMailBoxInfo

メールボックス情報設定

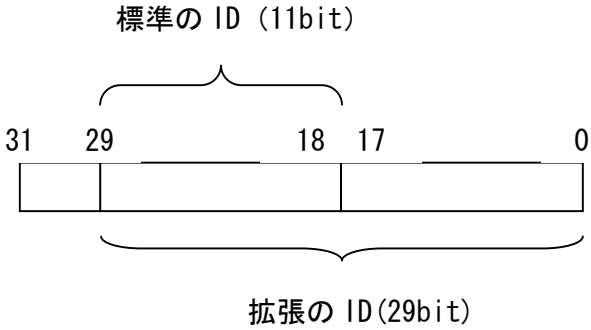
書式

```
signed short MyCanSetMailBoxInfo (
  unsigned long id,
  unsigned char ide,
  unsigned char direction,
  unsigned short interval,
  unsigned char timing,
  unsigned long maskID,
  unsigned char maskIde );
```

引数			
	unsigned long id	I	送信先 ID/受信元 ID （ide が 0（標準）の場合、1～7FFH） （ide が 1（拡張）の場合、1～1FFFFFFFH）
	unsigned char ide	I	拡張 ID（0：標準／1：拡張）
	unsigned char direction	I	方向（0：受信／1：送信）
	unsigned short interval	I	接続確認間隔（秒） ※0 の場合、接続確認を行いません。 ※接続確認用にリモートフレームを送信しますが、その際のデータ長は 8Byte 固定となります。
	unsigned char timing	I	送信タイミング （0：データ設定時に送信、 1：リクエスト要求時送信） ※リクエスト要求時は、リモートフレーム受信時に、自動的にデータ送信します。 ※マスク ID 及び拡張 IDE マスク設定時には、ID 及びマスク ID に対応したリモートフレームを受信した ID に対してデータを自動送信します。 ※方向が受信(0)の場合には、0 を設定してください。
	unsigned long maskID	I	マスク ID（0～1FFFFFFFH） マスク ID に対応するビットが取得可能になります。 本パラメータを設定することにより、送信先/受信元 ID 以外の ID を受信することが可能になります。 ※使用しない場合には、0 を設定ください。 （例） ID：3、マスク ID：6 の場合、以下の ID のデータ受信することが可能です。

ID	3	(=0011b)
マスク ID	6	(=0110b)
受信可能 ID	1	(=0001b)
	3	(=0011b)
	5	(=0101b)
	7	(=0111b)

	unsigned char maskIde	I	拡張 ID マスクフラグ （0：対応する拡張 ID が有効 1：対応する拡張 ID が無効） 本パラメータを設定することにより、拡張 ID が標準の場合でも拡張 ID のデータ、拡張 ID が拡張の場合でも標準 ID のデータを取得することが可能になります。 ※使用しない場合には、0 を設定ください ※本パラメータを 0 に設定した場合には、マスク ID は、送信先 ID/受信元 ID と同じ範囲になります。 ※本パラメータを 1 に設定した場合には、マスク ID は、拡張時の ID と同じ範囲になりますが、以下のような設定となりますので、注意が必要になります。



戻り値	OK(0)	：正常終了
	NG(-1)	：異常終了 ・ ID が範囲外 ・ 拡張 ID が範囲外 ・ 方向が範囲外 ・ 送信タイミングが範囲外 ・ マスク ID が範囲外 ・ 拡張マスクフラグが範囲外 ・ 設定可能個数を超過 ・ CAN ライブラリ状態が DISABLE 以外

解説

メールボックスに初期設定を行います。
設定順序がそのまま送受信の優先度になりますので、後から同一の ID を設定した場合、送受信できなくなる場合がございます。
対応マイコンによりメールボックスへの設定可能な個数が異なります。

- SH7216：13 個
- SH7262/7264：29 個
- H8S/2556: 13 個
- SH72A2: 62 個(送信設定最大数 31 個　受信設定最大数 62 個)

※ID=0 は本ライブラリのセキュリティ用に使用しており、maskID による ID=0 も取得できません。
※H8S/2556 の場合、マスク設定不可能となります。
「maskID」,「maskIde」については設定値は Don’Care となります。

MyCanEnable

CAN 通信実行

書式	signed short MyCanEnable (void)	
引数	なし	
戻り値	OK(0)	: 正常終了
	NG(-1)	: 異常終了
		・ CAN ライブラリ状態が DISABLE 以外
解説	CAN 通信を開始します。	

MyCanDisable

CAN 通信停止

書式	signed short MyCanDisable (void)	
引数	なし	
戻り値	OK(0)	: 正常終了
	NG(-1)	: 異常終了
		・ CAN ライブラリ状態が ENABLE 以外
戻り値	なし	
解説	CAN 通信を停止します。	

MyCanSetSendData

データ送信

書式

```
signed short MyCanSetSendData (
  unsigned long id,
  unsigned char ide,
  unsigned long maskID,
  unsigned char maskIde,
  unsigned char *pData,
  unsigned char length )
```

引数			
	unsigned long id	I	送信先 ID
	unsigned char ide	I	拡張 ID（0：標準／1：拡張）
	unsigned long maskID	I	マスク ID
	unsigned char maskIde	I	拡張 ID マスクフラグ
	unsigned char *pData	I/O	送信データバッファポインタ
	unsigned long length	I	送信データサイズ(1～8)

戻り値

OK(0)　：正常終了

NG(-1)　：異常終了

- 送信先 ID がMyCanSetMailBoxInfo関数で設定されていない値である場合
- 拡張 ID がMyCanSetMailBoxInfo関数で設定されていない値である場合
- マスク ID がMyCanSetMailBoxInfo関数で設定されていない値である場合
- 拡張 ID マスクフラグがMyCanSetMailBoxInfo関数で設定されていない値である場合
- 送信データバッファが NULL
- 送信データ長が範囲外
- CAN ライブラリ状態が ENABLE 以外

解説

指定されたデータをデータサイズ分、メールボックスに設定し、送信タイムアウト後に送信します。

MyCanGetSendDataStatus関数をコールし“SENDING”でないことを確認してから、本関数を実行してください。

※H8S/2556 の場合、マスク設定不可能となります。
「maskID」,「maskIde」については設定値は Don’Care となります。

MyCanGetSendDataStatus

データ送信状態取得関数

書式	signed short MyCanGetSendDataStatus (void)	
引数	なし	
戻り値	SEND_FINISH(1)	：初期状態/送信完了
	SENDING(2)	：送信中
	NG(-1)	：異常終了
		・ CAN ライブラリ状態が INIT
解説	データ送信状態を返します。	
	データ送信時には、本関数をコールし“SENDING”ではないことを確認してから MyCanSetSendData関数コールしてください。	

MyCanGetRecvData

データ受信

書式

```
signed short MyCanGetRecvData (
  unsigned long *pId,
  unsigned char *pIde,
  unsigned long maskID,
  unsigned char maskIde,
  unsigned char *pData );
```

引数	unsigned long *pId	I/O	受信元 ID
	unsigned char *pIde	I/O	拡張 ID（0：標準／1：拡張）
	unsigned long maskId	I	マスク ID
	unsigned char maskIde	I	拡張 ID マスクフラグ
	unsigned char *pData	I/O	受信データバッファポインタ

戻り値

受信データサイズ

以下の場合には、「0」が返ります。

- 受信データがない場合

以下の場合には、「-1」が返ります。

- 受信元 ID がMyCanSetMailBoxInfo関数で設定されていない値である場合
- 拡張 ID がMyCanSetMailBoxInfo関数で設定されていない値である場合
- マスク ID がMyCanSetMailBoxInfo関数で設定されていない値である場合
- 拡張 ID マスクフラグがMyCanSetMailBoxInfo関数で設定されていない値である場合
- 受信バッファポインタが「NULL」
- CAN ライブラリ状態が ENABLE 以外

解説

通信相手から送信されたデータを受信データバッファポインタに格納します。
メールボックス情報設定時（MyCanSetMailBoxInfo関数コール時）に、マスク ID を設定している場合、受信したデータの ID 及び拡張 ID が更新されます。

※本関数を呼び出すときには、アプリケーションで 8Byte を確保してから本関数をコールしてください。

※H8S/2556 の場合、マスク設定不可能となります。
「maskID」,「maskIde」については設定値は Don’Care となります。

MyCanGetStatus

CAN 状態取得

書式	unsigned short MyCanGetStatus (void)	
引数	なし	
戻り値	ERROR_ACTIVE(1)	: エラーアクティブ
	ERROR_PASSIVE(2)	: エラーパッシブ
	BUS_OFF(3)	: バスオフ
	NG (-1)	: 異常終了
		・ CAN ライブラリ状態が INIT
解説	CAN ユニットの状態を返します。	

MyCanGetLibStatus

CAN ライブラリ状態取得

書式	unsigned char MyCanGetLibStatus (void)	
引数	なし	
戻り値	STATUS_INIT(0)	: 初期状態
	STATUS_DISABLE(1)	: CAN 通信無効状態
	STATUS_ENABLE(2)	: CAN 通信有効状態
解説	CAN ライブラリ状態を返します。	

MyCanSetPriority

割り込み優先度設定

書式	signed short MyCanSetPriority (void)	
引数	unsigned short priority	割り込み優先度(レベル : 0～15)
戻り値	OK(0)	: 正常終了
	NG (-1)	: 異常終了
		・ CAN ライブラリ状態が DISABLE 以外
戻り値	なし	
解説	指定された CAN の各割り込み優先度を設定します。 15 以上の値を設定した場合には、「15」を設定します。 ※MyCanInit関数をコールした場合には「15」が設定されますので、割り込み優先度を変更する場合には、MyCanInit関数をコールした後に本関数をコールしてください。 ※H8S/2556 の場合、 割り込み優先度が 0（最低）～7（最高）となります。 7 以上の値を設定した場合には、「7」を設定します。	

MyCanInterruptRm

データフレーム／リモートフレーム受信割り込み

書式	void MyCanInterruptRm (void)
引数	なし
戻り値	なし
解説	データフレーム／リモートフレーム受信割り込み処理を行います。 ・ メールボックスに格納されているデータをデータバッファに格納 ・ リクエスト要求時にデータを送信する場合、リモートフレーム受信時にメールボックスに格納済みデータを送信 ※本関数を割り込みベクターテーブルに登録してください。 API 関数ではございませんので、アプリケーションからの関数コールは行わないでください。 ※H8S/2556 の場合、 データフレーム／リモートフレーム送信完了割り込み処理も行います。

MyCanInterruptSIe

データフレーム／リモートフレーム送信完了割り込み

書式	void MyCanInterruptSIe (void)
引数	なし
戻り値	なし
解説	データフレーム／リモートフレーム送信完了割り込み処理を行います。 ・ データフレーム／リモートフレーム送信時に送信ステータスを送信完了に設定 ※本関数を割り込みベクターテーブルに登録してください。 API 関数ではございませんので、アプリケーションからの関数コールは行わないでください。 ※H8S/2556 の場合、未定義関数となります。

MyCanInterruptTimer

タイマー割り込み

書式	void MyCanInterruptTimer (void)
引数	なし
戻り値	なし
解説	1ms のタイマーでの割り込み処理を行います。 メールボックス情報より、接続確認時間に到達するとリモートフレームを送信します。 対応マイコンにより割り込み要因が異なります。 ・ SH7216：CMI1 ・ SH7262/7264：OVRn（n：チャネル番号） ・ H8S/2556：TMU3 ・ SH72A2：CMI1 ※本関数を割り込みベクターテーブルに登録してください。 API 関数ではございませんので、アプリケーションからの関数コールは行わないでください。

8. 付録

CD に付属していますサンプルアプリケーションついて説明します。

サンプルアプリケーションの実行環境について、デバッガ未使用時のものを図 8に、デバッガ使用時のものを図 8に示します。

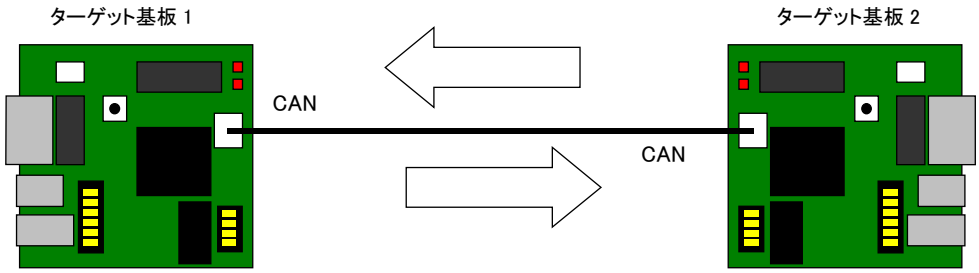


図 8.1　デバッガ未使用時のサンプルアプリケーションの実行環境

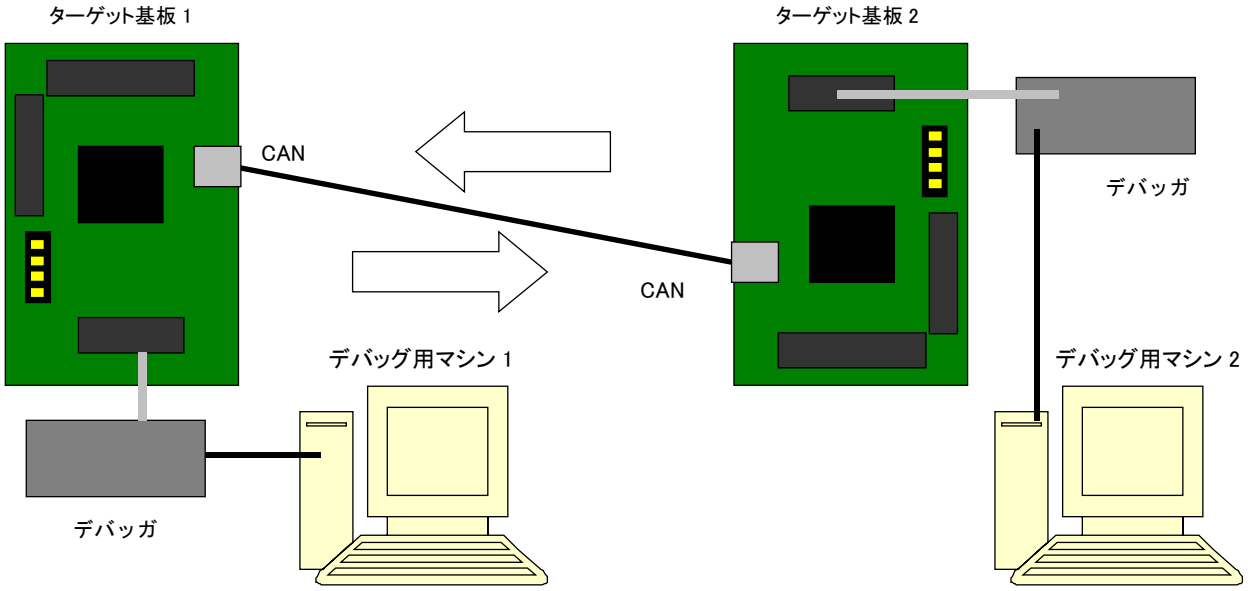


図 8.2　デバッガ使用時のサンプルアプリケーションの実行環境

サンプルアプリケーションは、ターゲット基板 1 とターゲット基板 2 で簡単な CAN 通信を行うものです。
サンプルアプリケーションを実行するための手順について、デバッガ未使用時とデバッガ使用時をそれぞれ以下に示します。

※対応マイコンにより、サンプルアプリケーション実行環境の対応が異なります
・ デバッガ未使用時のサンプルアプリケーション実行環境
SH7216

・ デバッガ使用時のサンプルアプリケーション実行環境
SH7262/7264

■デバッガ使用時の実行方法
(1) CD に入っています「sample1.mot」をお手持ちのライターでターゲット 1 に、「sample2.mot」をターゲット基板 2 に書き込んでください。
尚、sample1.mot は、「まい CANLite」をリンクして作成しており、sample2.mot は、「まい CAN」をリンクして作成しております。
「まい CAN」と「まい CANLite」の違いについては、9章をご参照ください。
(2) ターゲット基板 1 とターゲット基板 2 を CAN ケーブルで接続し、ターゲット基板の電源を投入してください。
(3) 各ターゲット基板の SW を押下すると、対向のターゲット基板の LED が点灯します。

■デバッガ未使用時の実行方法
(1) CD に入っています「sample/sample1」、「sample/sample2」をそれぞれデバッグ用マシンにコピーしてください。
尚、「sample/sample1」は、「まい CANLite」をリンクして作成しており、「sample/sample2」は、「まい CAN」をリンクして作成しております。
「まい CAN」と「まい CANLite」の違いについては、9章をご参照ください。
(2) 各デバッグ用マシンと各ターゲット基板を E10A-USB にて接続してください。
(3) 各デバッグ用マシンにて (1) でコピーした HEW プロジェクトを起動し、ターゲットファイルをダウンロードしてから実行してください。
(4) ターゲット基板 1 とターゲット基板 2 を CAN ケーブルで接続し、ターゲット基板 2 の電源を投入してください。
(5) HEW から「SEND_DATA」を変更すると、対向のターゲットを実行している HEW の「RECV_DATA」の値が更新されます。

9. 「まい CAN」と「まい CANLite」の違いについて

「まい CANLite」は、「まい CAN」とは異なり、データ送受信サイズが 1Byte に固定されます。「まい CAN」をご購入のお客様は、「まい CAN」を使った動作確認用に「まい CANLite」(mycan_0100_xxx.lib)を「sample」フォルダに格納しておりますので、ご利用ください。

10. 著作権・免責について

本書の内容は予告無く変更する場合があります。本書は著作権により保護されています。
株式会社北斗電子(以下当社)の文書による事前の許諾無しに、本書を複写または複製、転載する事は禁じられています。

当社は本書の内容について万全を期して作成し、正確と確信しておりますが、当社による本書に関する保証は一切なく、特定の目的の市販性、正当性、適合性に関する黙示の保証に対する責任を否認します。本書の内容は予告無しに変更する事があります。本書の中に誤りがある場合でも、当社はいかなる責任も負いません。

本書の内容に関するお問い合わせはご容赦下さい。

本ソフトウェアをご利用になる際、別紙「ソフトウェア使用権許諾契約書」を熟読ください。

本ソフトウェアの価格にはサポート料は含まれておりません。また、現地調査等が必要な場合はサポート料＋交通費や宿泊費がかかります。

最新情報については弊社ホームページをご活用下さい。

URL:http://www.hokutodenshi.co.jp

まい CAN　取扱説明書

© 2011-2012 北斗電子 Printed in Japan 2011 年 1 月 13 日初版発行　REV.1.0.4.0 (121127)

発行　株式会社北斗電子
〒060-0042 札幌市中央区大通西 16 丁目 3 番地 7
TEL：011-640-8800
FAX：011-640-8801
e-mail：support@hokutodenshi.co.jp（サポート用）
order@hokutodenshi.co.jp（ご注文用）
URL:http://www.hokutodenshi.co.jp